MIDI Media Adaptation Layer for IEEE-1394

MMA/AMEI RP-027

Version 1.0
November 30, 2000

Published By:
The Association of Musical Electronics Industry
Tokyo, Japan

The MIDI Manufacturers Association
Los Angeles, CA

This specification ("MIDI Media Adaptation Layer for IEEE-1394") is a collaborative work done by the Association of Musical Electronics Industry (AMEI) in Japan and the MIDI Manufacturers Association (MMA) outside of Japan. It is the recommendation of these two bodies, representing the MIDI industry world-wide, that this specification be followed for the transmission and receipt of MIDI messages using IEEE-1394.

This specification has been referenced by the 1394 Trade Association, in 1394TA Document 1999014 “Enhancement to Audio and Music Data Transmission Protocol 1.0”, July 10, 2000. This specification is also referenced in the forthcoming edition of IEC 61883-6 “Consumer Audio/Video Equipment – Digital Interface – Part 6: Audio and music data transmission protocol”, International Electrotechnical Commission.

The MMA, AMEI, and all companies involved in the development of this specification shall not be held liable to any person or entity for any reason related to the adoption, implementation or other use of this document or the specification herein.

Revision History

This specification was initially approved at the Annual General Meeting of the MIDI Manufacturers Assocation, February 6, 2000, and then amended jointly by MMA and AMEI working group members. The 1.0 version of this specification was approved by the MMA membership on March 20, 2000. Subsequently, explanatory text was added to section [5] by the MMA Technical Standards Board, regarding the nature and use of RID (IEEE Registration Authority Identifiers).

The following list summarizes the key differences between version 1.0 and version 0.75 (the initial version approved at the 2000 MMA Annual General Meeting.

- Changed **'should**' to '**shall**' in [1.2.4] Support for the MPX-MIDI Data Channel capability is now mandatory.
- Definition restored for 'MIDI Conformant Data Channel' in [1.4.3]
- Discussion of MPX-MIDI Channel moved to [3] (now normative)
- Annex A.3 restructured.

- Changed 'Ver.' to 'Version' throughout document
- Capitalized 'Cluster' everywhere for consistency (but not "event cluster").
- Explicit 'MIDI' prefix was used to describe 'MIDI Channel' in all places.
- Clarified language and definitions in a number of locations.
- Clarified transmission rate limitation for MIDI1.0—SPEED
- Revised capitalization of several terms for consistency.

- Added text to [5] clarifying nature and use of RID (Registration Authority ID, also known as an IEEE company_id or Organizationally Unique Identifier).

Copyright © 2000 Association of Musical Electronics Industry (Japan).
Copyright © 2000 MIDI Manufacturers Association Incorporated (outside Japan).

ALL RIGHTS RESERVED. NO PART OF THIS DOCUMENT MAY BE REPRODUCED OR TRANSMITTED IN ANY FORM OR BY ANY MEANS, ELECTRONIC OR MECHANICAL, INCLUDING INFORMATION STORAGE AND RETRIEVAL SYSTEMS, WITHOUT PERMISSION IN WRITING FROM THE COPYRIGHT HOLDERS.

MMA/AMEI RP-027
Printed 2000

MIDI Manufacturers Association Incorporated
PO Box 3173
La Habra CA 90632-3173

Specification for MIDI Media Adaptation Layer for IEEE1394
Version 1.0

November 30, 2000

Contents

[1] Introduction 2
- [1.1] Purpose of this document 2
- [1.2] Conformance Levels 2
- [1.3] References 2
- [1.4] Terminology 3
  - [1.4.1] IEEE1394 3
  - [1.4.2] MIDI 3
  - [1.4.3] Audio and Music Data Transmission Protocol 4

[2] Overview of MIDI Media Adaptation Layer for IEEE1394 5

[3] Encapsulation of MIDI data stream 5
- [3.1] Implementation of Encapsulation 6
  - [3.1.1] Transmission Rate Limitation 6
  - [3.1.2] MIDI1.0 –SPEED 6
  - [3.1.3] MIDI1.0 –2X–SPEED 7
  - [3.1.4] MIDI1.0 –3X–SPEED 7

[4] AM824 Data Channel configuration 7

[5] Unit information 8

[6] Appendix (informative) 10
- [6.1] Gateway 10

Annex A: General Overview (informative) 11
- A.1 Overview: Common Isochronous Packets and the AM Protocol 11
- A.2 Timing and Time Stamps 12
- A.3 Streams, Channels, Sequences and Clusters 13
  - A.3.1 Clusters and MPX-MIDI Data Channels 14
  - A.3.2 Timing for Audio and MIDI Data 15
  - A.3.3 Bandwidth 16

Figure 2.1 Overview of MIDI media adaptation layer 5
Figure 4.1 One "Cluster" ("Data Block") with CLUSTER_DIMENSION = 6 8
Figure 5.1 Unit directory for MIDI capable node 8
Figure 5.2 CSR-ROM example 9
Figure 5.4 Definition of MIDI registers 9
Figure 6.1 MIDI Port and MIDI-PLUG 10
Figure A.1 Basic CIP Structure (two-quadlet CIP header form) 11
Figure A.2 CIP Header 11
Figure A.3 Sequences and Cluster Events 14
Figure A.4 Clusters, AM824 Data Channels and MPX-MIDI Data Channels 15
Figure A.5 Nominal MIDI Timing Accuracy 16
Figure A.6 CIP Containing Multiple Clusters of AM824 Data 16

[1] Introduction

[1.1] Purpose of this document

This document defines the Specification for the MIDI Media Adaptation Layer for IEEE1394, on behalf of the MIDI Manufacturers Association (MMA) and the Association of Musical Electronics Industry (AMEI), as a Recommended Practice to implement the MIDI specification for IEEE1394.

Working Members
Technical Editors: Jim Wright, Yoshiharu Honjo, Jun-ichi Fujimori
Discussion Groups: MPWG of AMEI chaired by Yoshiharu Honjo (honjo@roland.co.jp).
TLWG of MMA chaired by Jim Wright (jwright@watson.ibm.com).

[1.2] Conformance Levels

[1.2.1] expected: A key word used to describe the behavior of the hardware or software in the design models assumed by this Specification. Other hardware and software design models may also be implemented.

[1.2.2] may: A key word that indicates flexibility of choice with no implied preference.

[1.2.3] shall: A key word indicating a mandatory requirement. Designers are required to implement all such mandatory requirements.

[1.2.4] should: A key word indicating flexibility of choice with a strongly preferred alternative. The phrase *it is recommended* has the same meaning.

This document defines the MIDI Media Adaptation Layer for IEEE1394, which includes:

- Encapsulation Details for transport of MIDI data streams by using isochronous transmission based on "Audio and Music Data Transmission Protocol Version 1.0" and "Enhancement to Audio and Music Data Transmission Protocol Version 1.0"
- Mechanism for multiplexing MIDI data streams using Data Block Count (*DBC*)
  This multiplexing is newly defined in "Enhancement to Audio and Music Data Transmission Protocol Version 1.0". All devices compliant with this specification shall have multiplexing capability and shall provide a descriptor for this capability as described in Unit information.

[1.3] References

"Complete MIDI 1.0 Detailed Specification", Version 96.1, MIDI Manufacturers Association, March 1996

"Complete MIDI 1.0 Detailed Specification", Japanese Version 98.1, Association of Musical Electronics Industry, 1998 (This document is a translation of the preceding reference.)

"TA1997001, Audio and Music Data Transmission Protocol Version 1.0", 1394 Trade Association, 1997 (known as IEC61883-6 PAS)

"TA1999014, Enhancement to Audio and Music Data Transmission Protocol Version 1.0 ", 1394 Trade Association, 1999

"IEEE1394-1995, Standard for a High Performance Serial Bus", IEEE, 1995

“IEEE1212-2000, Control and Status Register (CSR) Architecture for Microcomputer Busses”
(Draft standard can be downloaded from http://www.zayante.com/p1212r)

"IEC61883-1, Digital Interface for consumer electronics audio/video equipment - Part 1: General"

"MIDI Over Alternate Media Transports", Version 0.4c, MIDI Manufacturers Association, 1999

[1.4] Terminology

[1.4.1] IEEE1394

A number of terms from various IEEE1394 specification documents are used in this document. For informational purposes, brief explanations of these terms are given below. However, the actual definitions of these terms are contained in the referenced IEEE1394 specification documents.

“quadlet”: a 32-bit data element. Quadlets use big-endian byte ordering: byte 0 is always the most significant byte of the element. The field on the left (most significant) is transmitted first; within a field the most significant (leftmost) bit is also transmitted first. This ordering convention is shown below:

MSB	(bytes within a quadlet)		LSB
byte_0	byte_1	byte_2	byte_3

"data sequence": a series of aligned quadlets conforming to a particular format. Also known as "sequence." A data sequence is normally a continuous or connected series of related events, presented in time order. However, a data sequence may also be constructed by interleaving two or more multiplexed streams of events in a prescribed manner, so as to maintain time order and relevant structural constraints.

"Channel": a logical connection through which a particular data sequence or stream is transmitted. The term "Channel" is used here in compliance with standard IEEE1394 usage, and should not be confused with the term "MIDI channel" (See Annex A).

"CIP": Common Isochronous Packet. The basic IEEE1394 packet type used for transport of isochronous data such as video, audio and MIDI. The CIP is fully defined in IEC61883-1. A CIP overview is provided in Annex A. A partial list of CIP fields includes *SID* (source node of transmitter), *DBS* (data block size), *DBC* (data block count), *FMT* (format ID), *FDF* (format dependent field) and *SYT* (time stamp, optional).

"*SYT*": the field within the CIP header that specifies the time when the specified event within the CIP is to be presented at a receiver.

“Data Block”: a significant unit of data within a CIP (or other IEEE1394 packet). A CIP may contain 0, 1 or more Data Blocks. See discussion in Annex A.

"*DBC*": Data Block Count, an index number for a Data Block within a particular data sequence. The CIP header has a *DBC* field (8 bits) which contains the Data Block Count value for the first Data Block in that CIP (just after the CIP header).

"CSR": Control and Status Register defined in ISO/IEC 13213 (ANSI/IEEE 1212).

[1.4.2] MIDI

"MIDI message": a sequence of one or more bytes that constitutes a single MIDI event. Message types include Channel Voice, Channel Mode, System Common, System Real-Time and System Exclusive, and are defined in the "Complete MIDI 1.0 Detailed Specification" and subsequent MMA publications. The structure of MIDI messages is as follows.

One byte Message	Status Byte
Two byte Message	Status Byte - Data Byte
Three byte Message	Status Byte - two Data Bytes
System Exclusive Message	Status Byte - Data Bytes - EOX

Status bytes are eight-bit data with the MSB = 1 and they identify the message type. One byte messages consist of a single Status byte only. All other messages consist of a Status byte followed by one or more Data bytes that carry the

content of the message. Data bytes are eight-bit data with the MSB = 0. A System Exclusive message can contain any number of Data bytes, and is terminated by an End of Exclusive (EOX) Status byte.

The following terms are defined within “MIDI Over Alternate Media Transports”, but are not part of the “Complete MIDI 1.0 Detailed Specification”.

"MIDI 1.0 Application Layer": the set of application-level byte stream protocols used to access services provided by MIDI devices, independent from the actual means of message packaging, delivery and transmission. Also known as ‘MIDI 1.0 Logical Layer’. Application-layer services include Channel, System and System Exclusive message transport; Sample Dump, File Dump, MIDI Tuning and Device Inquiry; MIDI Time Code, MIDI Show Control and MIDI Machine Control; General MIDI and Downloadable Sounds; and other services as defined in the Complete MIDI 1.0 Detailed Specification and subsequent documents.

“Media Adaptation Layer”: the layer responsible for translating media data from its native format to and from the format required by a given transport layer.

“MIDI 1.0 Transport Layer”: the standard media transport used for delivery of MIDI messages and MIDI streams. The MIDI 1.0 transport layer implements an isochronous unreliable simplex connection-oriented point-to-point media transport on top of the asynchronous serial MIDI 1.0 Physical Layer.

"MIDI 1.0 Physical Layer": the standard physical link used for MIDI message transport. The MIDI 1.0 physical layer is implemented as an asynchronous serial transmission line using an opto coupler and 5-pin DIN connector and operating at 31.25 kbps (31,250 bits per second), as defined in the Complete MIDI 1.0 Detailed Specification. The MIDI 1.0 physical layer provides isolated grounds to avoid potential audio degradation (e.g. power line hum) and safety hazards.

"MIDI 1.0 data stream": a byte stream that complies with the MIDI 1.0 Application Layer protocols. The transmission rate for such streams is limited to 31.25kbps regardless of the underlying Physical or Transport Layer.

"MIDI data stream": a byte stream that consists of MIDI messages. Note: when the term "MIDI data stream" is used (with the version number "1.0" intentionally removed), it denotes a data stream which may not be limited to a transmission rate of 31.25 kbps.

[1.4.3] Audio and Music Data Transmission Protocol

The following terms are specific to the Audio and Music Data Transmission Protocol and/or this document.

"Cluster": an ordered collection of events that occur at the same time. When several event sequences are synchronized, it is possible to interleave these sequences into a single sequence consisting of "Clusters".

"CLUSTER_DIMENSION": the number of sequences combined together to form a particular Cluster.

"AM824": a 32-bit data element consisting of an 8-bit LABEL and a 24-bit data field. The basic data element used for encapsulating MIDI data within the proposed Audio and Music format.

"LABEL": denotes the format for a particular AM824 data element (8 bits).

"AM824 Data Channel": A Channel in which one type of AM824 data sequence is being transmitted.

"AM824 Data Channel Number": ordinal index of an AM824 Data Channel within a particular Cluster. The value 0 denotes the first AM824 Data Channel within the Cluster. (See [4] "AM824 Data Channel Configuration")

"MIDI Conformant Data Channel": An AM824 Data Channel in which one or more MIDI data streams are being transmitted. The AM824 LABELs for “MIDI Conformant Data” are shown below. Only these LABELs may be used within a MIDI Conformant Data Channel:

LABEL	Valid data size
80H	No Data = 000000H
81H	1 byte
82H	2 bytes
83H	3 bytes

(Note: a 2-digit number followed by an 'H' means a hexadecimal value.)

"MPX-MIDI Data Channel": a discontiguous sequence of AM824 elements used to carry a particular MIDI data stream. A number of MPX-MIDI Data Channels are multiplexed together within a single MIDI Conformant Data Channel. MPX-MIDI Data Channels are used only inside of the IEEE1394 media adaptation layer.

"*SFC*": Nominal Sampling Frequency Code. This element is part of the *FDF* field for a CIP with *FMT* = "Audio and Music Data". *SFC* also specifies the value for *SYT_INTERVAL*, as shown below:

For Fs = 32kHz, 44.1 kHz, 48 kHz: *SYT_INTERVAL* = 8
For Fs = 96 kHz: *SYT_INTERVAL* = 16
For Fs = 192 kHz: *SYT_INTERVAL* = 32
Refer to "Enhancement to Audio and Music Data Transmission Protocol Version 1.0" for details.

[2] Overview of MIDI Media Adaptation Layer for IEEE1394

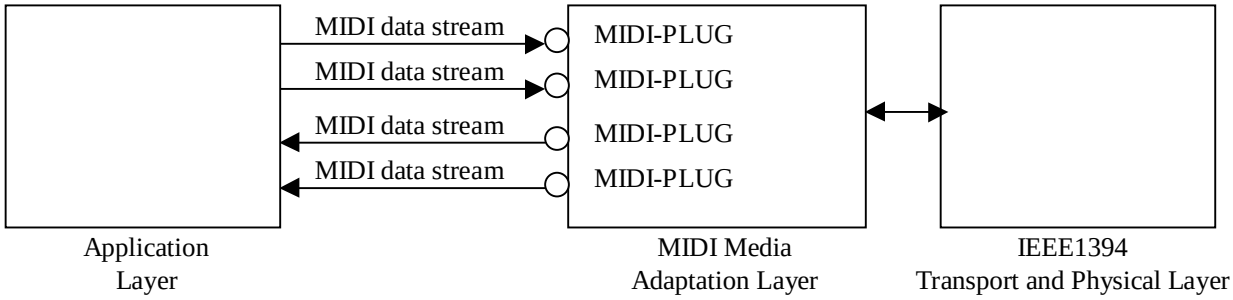

Figure 2.1 Overview of MIDI media adaptation layer

From an application-layer viewpoint, the IEEE1394 MIDI media adaptation layer has zero, one or more input/output end points for MIDI data streams. An end point will be called a "MIDI-PLUG" for convenience (a “MIDI-Plug” is equivalent to a MIDI Endpoint as defined in "MIDI Over Alternate Media Transports"). Each MIDI-PLUG carries one input MIDI data stream or one output MIDI data stream.

The application-layer entity, which will utilize the IEEE1394 media adaptation layer, determines the mapping of MIDI data streams to MIDI-PLUGs. The specification of this mapping is outside the scope of this document.

[3] Encapsulation of MIDI data stream

A MIDI data stream is transmitted in a “byte transparent” manner: i.e., no data modification or loss shall be allowed in a Media Adaptation Layer.

The purpose of encapsulation is to package individual MIDI messages into 1394-compatible format in a way that supports simple, reliable parsing by the receiver. This work is performed by an entity in the media adaptation layer known as the ***encapsulation function block.*** This entity is also responsible for converting the encapsulated data from 1394-compatible format back into MIDI data stream format.

According to the multiplexing mechanism defined in “Enhancement to Audio and Music Data Transmission Protocol Version 1.0,” the encapsulation function block shall multiplex up to 8 MIDI data streams into a single MIDI Conformant Data Channel. Each successive AM824 Data Element comprising the MIDI Conformant Data Channel shall carry a particular MIDI data stream ‘M’, where

$$M = \mathrm{mod}(DBC, 8)$$

The set of potential MIDI data streams is thereby numbered successively from 0 to 7. Each such multiplexed MIDI data stream is carried by a distinct discontiguous sequence of AM824 data elements. Each such sequence of AM824 data elements is referred to as an MPX-MIDI Data Channel, as defined above.

[3.1] Implementation of Encapsulation

NOTE: a future extension of this specification will define a negotiation procedure between transmitters and receivers and will allow the use of LABEL 82H and 83H quadlets. These capabilities will allow transmission of MIDI data at rates substantially faster than the MIDI 1.0 data rate. However, the current version of this specification limits the use of these capabilities in order to avoid interoperability problems with other MIDI devices. Higher speed modes (e.g. MIDI1.0–2X–SPEED and MIDI1.0–3X–SPEED) are described solely so that devices compliant with the current specification will be able to interoperate with devices compliant with a future extension of this specification.

[3.1.1] Transmission Rate Limitation

The encapsulation function block of a transmitter that does not implement a negotiation procedure shall use a maximum rate of one LABEL 81H quadlet every 320 microseconds per MPX-MIDI Data Channel, regardless of the sampling frequency specified by SFC.

NOTE: After a negotiation procedure is defined, the MPX-MIDI Data Channel rate may be determined by the negotiation procedure.

There is no transmission rate limitation for LABEL 80H 'No Data' quadlets. The transmitter should send a LABEL 80H quadlet whenever there is no actual MIDI data to send (or when transmission of MIDI data must be delayed in order to comply with the transmission rate limitation). Note: if the CIP contains only MIDI Conformant data and there is no actual MIDI data to send, the CIP should be an empty packet rather than a packet containing only 'No Data' quadlets.

Timing accuracy (see Annex A.3) is implementation dependent.

[3.1.2] MIDI1.0 –SPEED

An MPX-MIDI Data Channel operating in MIDI1.0–SPEED mode shall use only LABEL 80H and 81H quadlets. The encapsulation function block of a transmitter operating in MIDI1.0-SPEED mode shall use a maximum MPX-MIDI Data Channel rate of one data byte every 320 microseconds, even though the available bandwidth may be greater.

The encapsulation function block of a receiver operating in MIDI1.0–SPEED mode shall stop receiving if it receives a MIDI Conformant data element with LABEL 82H or 83H. In this case, the receiver should also notify the application.

A transmitter or receiver that does not implement a negotiation procedure shall use only this MIDI1.0–SPEED mode. A transmitter or receiver that implements a negotiation procedure should use the MIDI1.0–SPEED by default. Such a transmitter shall not send data at a rate faster than one MIDI data byte every 320 microseconds **unless** the transmitter can confirm, through use of a negotiation procedure, that all receivers configured to receive that Data Channel can accept the faster data rate. In this case, that Data Channel can no longer be described as MIDI1.0-SPEED.

[3.1.3] MIDI1.0 –2X–SPEED

An MPX-MIDI Data Channel operating in MIDI1.0–2X–SPEED mode shall use only LABEL 80H, 81H and 82H quadlets. This supports a data rate of up to two bytes every 320 microseconds (or faster if a higher Data Channel rate is determined using the negotiation procedure).

The encapsulation function block of a receiver operating in MIDI1.0–2X–SPEED mode shall stop receiving if it receives a MIDI Conformant data element with LABEL 83H. In this case, the receiver should also notify the application.

MIDI1.0–2X–SPEED mode shall not be used until the negotiation procedure and all encapsulation details for MIDI1.0–2X–SPEED mode are defined in a future extension of this specification.

As noted above, reception of a LABEL 82H quadlet by a receiver operating in MIDI1.0–SPEED mode shall cause the encapsulation function block of that receiver to stop receiving.

[3.1.4] MIDI1.0 –3X–SPEED

An MPX-MIDI Data Channel operating in MIDI1.0–3X–SPEED mode shall use only LABEL 80H, 81H, 82H and 83H quadlets. This supports a data rate of up to three bytes every 320 microseconds (or faster if a higher Data Channel rate is determined using the negotiation procedure).

MIDI1.0–3X–SPEED mode shall not be used until the negotiation procedure and all encapsulation details for MIDI1.0–3X–SPEED mode are defined in a future extension of this specification.

As noted above, reception of a LABEL 82H or 83H quadlet by a receiver operating in MIDI1.0–SPEED mode shall cause the encapsulation function block of that receiver to stop receiving.

[4] AM824 Data Channel configuration

A CIP containing actual AM824 data includes one or more "Clusters", also known as "cluster events". (A CIP conforming to AM824 subformat can also be an "empty" packet, or may consist entirely of NO-DATA elements.)

Each such Cluster consists of CLUSTER_DIMENSION number of AM824 Data elements. Each AM824 Data element is associated with a specific AM824 Data Channel. The Isochronous Channel associated with a single such Cluster has CLUSTER_DIMENSION number of AM824 Data Channels, zero or more of which may be a "MIDI Conformant Data Channel". The LABEL of each AM824 Data element defines the type of the corresponding AM824 Data Channel. AM824 Data Channels may be identified using ordinal numbers. Data Channel Number 0 corresponds to the first AM824 Data element within a given Cluster. Successive Data Channels are numbered in ascending monotonic order.

The encapsulation function block should define the configuration of AM824 Data Channels. An example configuration of AM824 Data Channels is shown in Figure 4.1. Note that each MIDI Conformant Data Channel can include up to 8 distinct MPX-MIDI Data Channels.

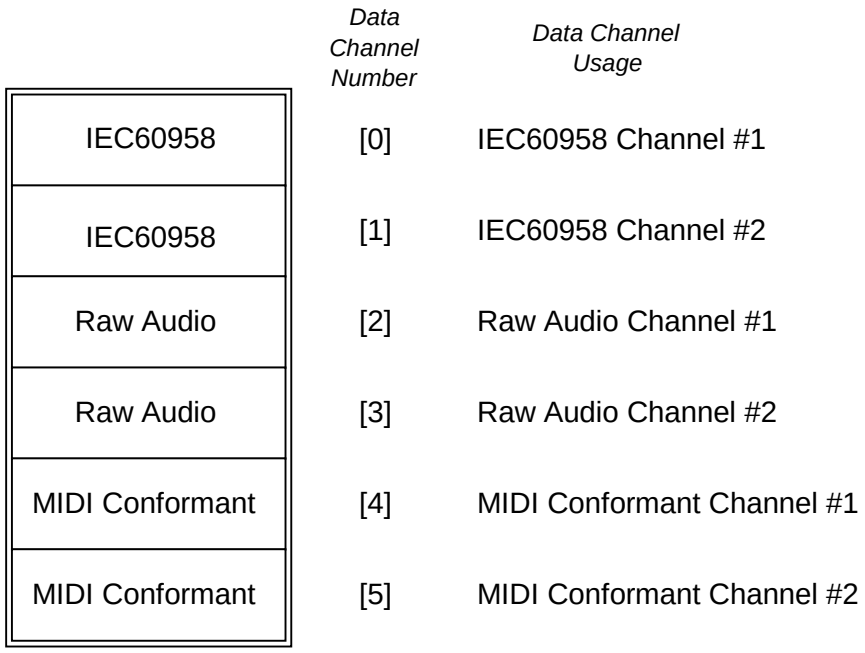

Figure 4.1 One "Cluster" ("Data Block") with CLUSTER_DIMENSION = 6

The encapsulation function block of a transmitter must associate each input MIDI-PLUG with a single MPX-MIDI Data Channel such that the index number (0,1, 2,...) of each input MIDI-PLUG is identical to the associated MPX-MIDI Data Channel number. When two or more MIDI Conformant Data Channels are used, the first set of input MIDI-PLUGs is associated with the lowest-numbered MIDI Conformant Data Channel. The remaining MIDI-PLUGs should be associated with the remaining MIDI Conformant Data Channels in ascending order.

The Isochronous Channel associated with a given input MIDI-PLUG may be arbitrarily chosen. However, it is recommended that a single Node should use only one Isochronous Channel for transmission in order to conserve Isochronous Channels.

The encapsulation function block of a receiver must have the capability of associating each output MIDI-PLUG with an arbitrary MPX-MIDI Data Channel belonging to an arbitrary Isochronous Channel.

[5] Unit information

A node which conforms to this specification shall provide the unit directory for this specification in the CSR-ROM.

Entry name	**Description**
Specifier_ID	00-01-F6$_{16}$ is the 24-bit RID (Registration Authority ID, assigned by IEEE/RAC)
Version	01$_{16}$ is the version number that indicates the node conforms to this specification
Dependent_info	This entry next to the Version entry specifies the offset to the MIDI-specific register space from the 64-bit private space (from FFFF FFFF E000 0000$_{16}$). This value may be changed after bus reset occurs.

Figure 5.1 Unit directory for MIDI capable node

An example of the CSR-ROM is shown in Figure 5.2. Note that two distinct Specifier_ID values are used in this example. The first Specifier_ID (in the Unit Directory for AV/C) identifies the 1394 Trade Association as the organization defining the Unit Directory. The second Specifier_ID (in the Unit Directory for MIDI) jointly

identifies the Association of Musical Electronics Industry and the MIDI Manufacturers Assocation as the organizations defining the Unit Directory for MIDI.

Note: a Specifier_ID is equivalent to an IEEE company_id or or Organizationally Unique Identifier (OUI), as defined within IEEE 1212-2000 (Control and Status Register (CSR) Architecture for Microcomputer Busses) and IEEE Std 1394-1995, clause 8.3.2.5, "Configuration ROM." The IEEE Registration Authority Committee assigns company_id/OUI values; see < http://standards.ieee.org/regauth/oui/index.shtml> for further information.

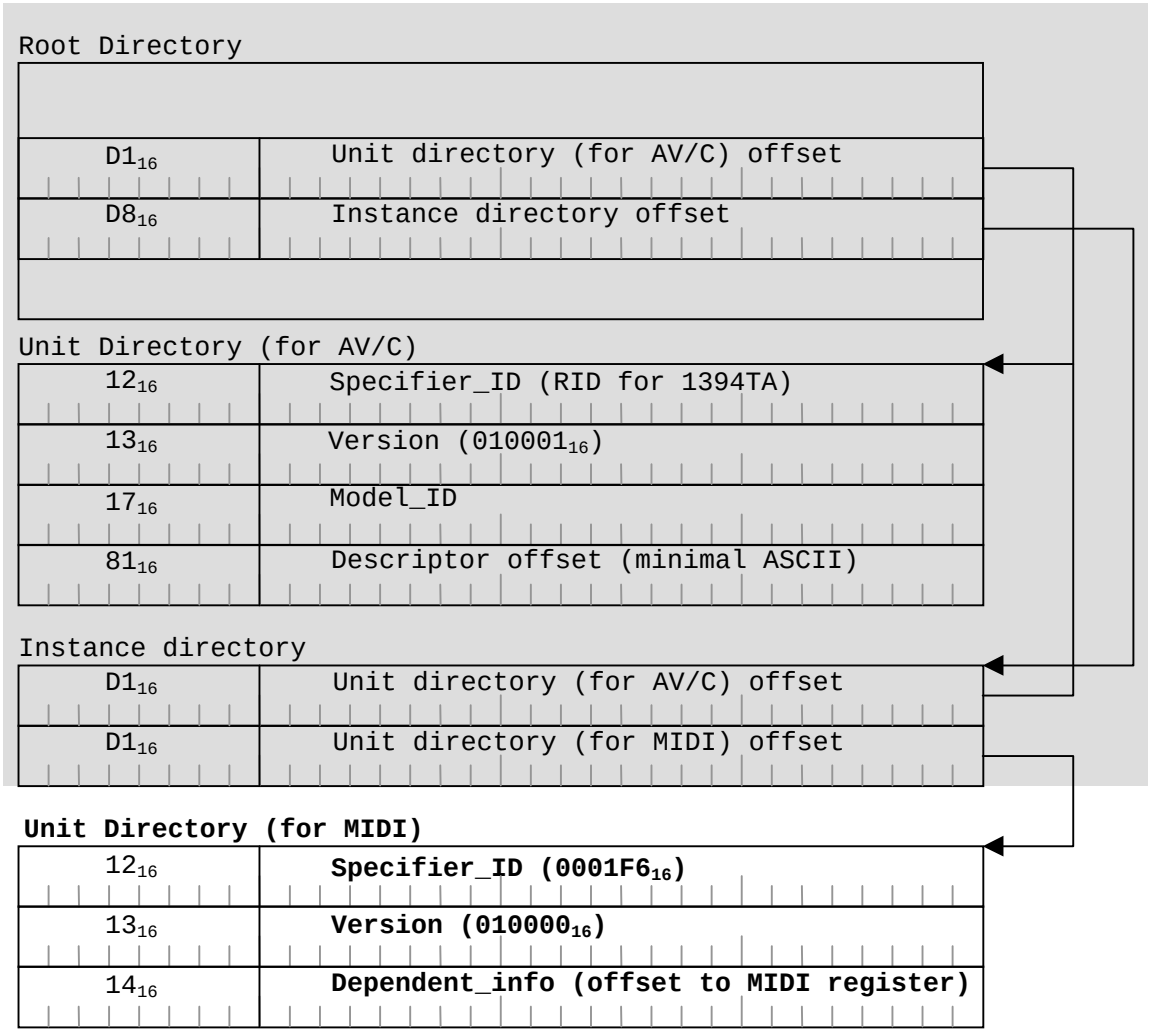

Figure 5.2 CSR-ROM example

Offset	BYTE1	BYTE2	BYTE3	BYTE4
00 00$_{16}$	Total byte size of registers (0)			

Figure 5.3 Definition of MIDI registers

The definition of MIDI-capabilities will be given in a future extension of this specification.

[6] Appendix (informative)

[6.1] Gateway

A MIDI port can be a MIDI 1.0 port, a software service accessed using some kind of application programming interface, or some other kind of hardware or software entity. Any further specification of MIDI Ports is outside the scope of this document.

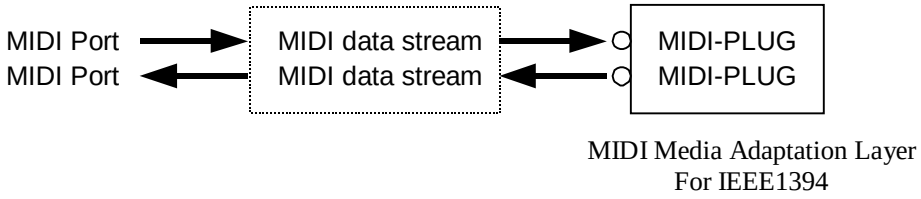

Figure 6.1 MIDI Port and MIDI-PLUG

Each MIDI-PLUG must be connected to a MIDI Port in order for external entities to transmit or receive MIDI data using 1394. The connection or "mapping" between a MIDI-PLUG and the corresponding MIDI Port may be fixed or changeable. If a 1394-MIDI Gateway or similar device can change such mappings, the following guidelines should be observed:

a) If a MIDI message (currently undefined) is used to change such mappings, it should be transmitted to a special MIDI-PLUG on the gateway device which is dedicated for carrying gateway command functions;
b) Such mappings may be changed using a method which is defined outside of the MIDI specification

Either or both methods may be supported by a gateway device.

Any further description of mapping change commands or other gateway command functions is outside the scope of this document.

Annex A: General Overview (informative)

This section is divided into three parts. The first part provides a brief overview of the Audio and Music Data Transmission Protocol ("AM Protocol") with a focus on MIDI-related aspects. The second part discusses timing and time stamps. The final part describes the relationships between various AM Protocol concepts (sequences, Clusters, streams and Channels), MIDI concepts (streams, MIDI channels), and the MPX-MIDI Channel mechanism. Normative definitions are provided in section [1.4] of this document and in the referenced base documents.

A.1 Overview: Common Isochronous Packets and the AM Protocol

The Audio and Music Data Transmission Protocol transports isochronous audio and MIDI data using Common Isochronous Packets. The basic CIP structure is shown in Figure A.1:

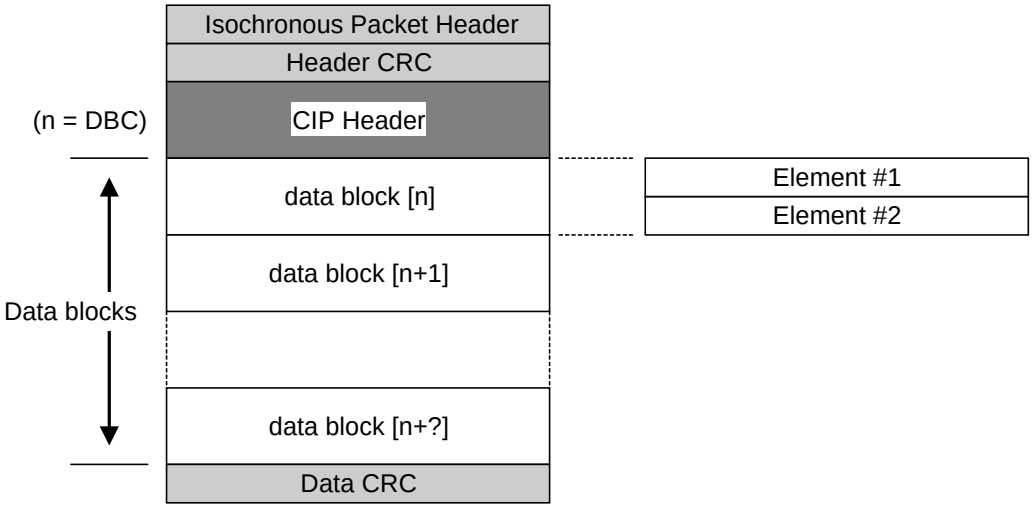

Figure A.1 Basic CIP Structure (two-quadlet CIP header form)

All fields are aligned on quadlet boundaries. The Isochronous Packet Header, Header CRC and Data CRC fields are common to all IEEE1394 isochronous packets (as defined by IEEE). The CIP format (defined in IEC61883-1) specifies an additional header and many other aspects.

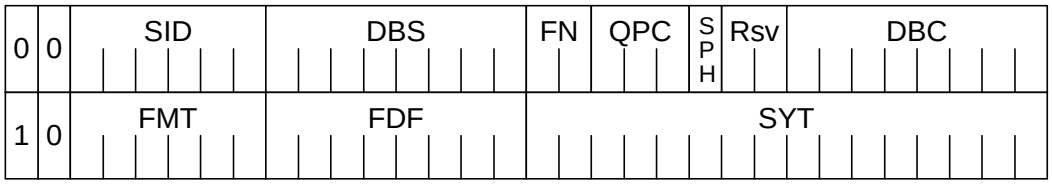

Figure A.2 CIP Header

The CIP Header contains a number of fields, including:

SID	Source ID of the transmitter originating the packet
DBS	Size of one Data Block (in quadlets); a CIP can contain 0, 1 or more Data Blocks
DBC	Data Block Count, normally the sequence number for the first Data Block in the CIP
FMT	Format ID: specifies DVCR, MPEG-2, AM Protocol…
FDF	Format Dependent Field; for AM Protocol, specifies subformat (see below)
SFC	Nominal Sampling Frequency Code (part of *FDF* for certain AM Protocol subformats)
SYT	Time stamp (optional), using low 16 bits of IEEE1394 CYCLE_TIME register
…	(other fields not described)

CIPs have a very flexible structure. However, the AM Protocol uses a somewhat restricted CIP format. Currently defined *FDF* values specify both the basic subformat (e.g. AM824, 24-bit * 4 Audio Pack, or 32-bit Floating Point) and the *SFC* value which indicates the nominal sample rate (e.g. 32 kHz, 44.1 kHz, 48kHz, 96 kHz).

The *SFC* value also specifies the *SYT_INTERVAL* value (normally 8; for 96 kHz, *SYT_INTERVAL* = 16). *SYT_INTERVAL* denotes the number of events between two successive valid *SYT*s (a single CIP may contain multiple events, but only a single *SYT*; also, *SYT* may be omitted from some CIPs).

A given CIP may contain only one *FMT* (e.g. only AM Protocol, or only MPEG-2). Furthermore, an AM Protocol CIP may contain only one subformat (e.g. AM824 or 32-bit Floating Point, but not both). Since AM824 data types include MIDI Conformant data, IEC 60958 audio data and Raw (16-24 bit) audio data, a single CIP may contain both MIDI Conformant data and one or more types of AM824 audio data.

A given CIP may contain one or more Data Blocks. All Data Blocks within a given CIP must be of the same size (specified by *DBS*). The contents of a given AM Protocol Data Block can be a basic data event (e.g. an AM824 quadlet or 32-bit floating point audio sample), a "pack" or a "Cluster." A *pack* is a set of non-32-bit aligned events, collected into a single aggregate for efficiency (e.g., four 24-bit audio samples packed into 3 quadlets). Packs are not used for MIDI Conformant data. A *Cluster* is a set of simple data events or pack events (but not both). Clusters are discussed further below. For the AM Protocol, all Data Blocks within a particular CIP must be of the same type (e.g. Cluster, pack or basic event). When a given CIP contains Clusters, there is exactly one Data Block for each Cluster in the CIP.

The AM Protocol reference specifications provide multiple definitions for a number of terms. Care should be taken to avoid confusing these multiple meanings:

- "data" has the normal meaning, but also means "an event which is neither a Cluster nor a pack."
- "event" can mean a basic event (e.g. AM824 quadlet), a "pack" or a "Cluster". While a Cluster is an event, a Cluster also contains events. (The contents of a pack are also called events.)

A.2 Timing and Time Stamps

This section is largely based on IEC61883-6, section 6.2. The next section provides additional information on timing of audio and MIDI data.

An AM Protocol CIP has only one time stamp: *SYT*. If the CIP contains multiple events (Data Blocks), it is necessary to specify which event within the CIP corresponds to the *SYT* time stamp. Time stamp resolution is 1/(24.576 MHz), or approximately 40 nanoseconds.

The transmitter prepares a time stamp for events that meet this condition:

mod(*data block count*, *SYT_INTERVAL*) = 0 [1]

data block count	running count of transmitted data blocks
SYT_INTERVAL	denotes number of events between two successive valid *SYT*s. This interval includes one event with a valid *SYT*. For example, if there are seven events between two successive valid *SYT*s, then *SYT_INTERVAL* = 8

The receiver determines which Data Block (cluster event, pack, basic event…) in the CIP corresponds to the *SYT*:

time stamp index = mod((*SYT_INTERVAL* – mod(*DBC*,*SYT_INTERVAL*), *SYT_INTERVAL)* [2]

time stamp index	denotes the Data Block (e.g. cluster event) with a time corresponding to *SYT*. 0-origin.
SYT_INTERVAL	(per above)
DBC	Data Block Count value for the first Data Block (e.g. cluster event) within the CIP

For example, assume that *DBC* = 6, *SYT_INTERVAL* = 8, and there are six cluster events (Data Blocks) in the CIP. In this case, *time stamp index* = mod((8- mod(6,8)), 8) = mod((8-6), 8) = 2, and *SYT* corresponds to the third cluster event. (See Figure A.6)

For AM Protocol format, every CIP header includes a *SYT* field. However, that *SYT* is not necessarily valid. If *time stamp index* denotes a Data Block which is not present within a given CIP, the *SYT* for that CIP is considered invalid, and the transmitter may set the *SYT* value to the "No Information" code (IEC61883-6).

The receiver is responsible for estimating the timing of events that occur between valid time stamps. The method for such estimation is implementation-dependent.

A.3 Streams, Channels, Sequences and Clusters

The MIDI specification supports partial message addressing through the use of 16 MIDI channels associated with a particular MIDI stream. This allows 16 distinct substreams of MIDI Channel Messages to be carried over a single MIDI stream. Other MIDI messages (e.g. System Common, System Real-time, System Exclusive) apply to the entire MIDI stream. Many current systems use multiple MIDI streams, to support more than 16 channels or to improve performance.

IEEE1394 supports much richer models for streams (and also for addressing). For IEEE1394, a ***stream*** consists of a time-ordered series of events (samples, messages or other data elements). A simple ***event sequence*** consists of a series of events from a single stream. When multiple event sequences are synchronized, it is possible to interleave the complete set of two or more synchronized event sequences into a single time-ordered sequence. That composite sequence contains a time-ordered set of ***Clusters*** (also known as *cluster events).*

Each Cluster consists of an ordered collection of one event (e.g. AM824 Data element) from each source sequence, as shown in Figure A.3. The number of sequences clustered together is called the ***dimension*** of the Cluster. (Note: when *pack* events are clustered together, the actual number of sequences is equal to CLUSTER_DIMENSION * UNIT_DIMENSION (the number of sequences per *pack*)). Each source sequence is carried by a corresponding Channel (logical connection). For AM824 elements, the LABEL of each AM824 Data element within a Cluster defines the type of the corresponding AM824 Data Channel. Additional information may be provided within the "unit directory" defined in the CSR architecture.

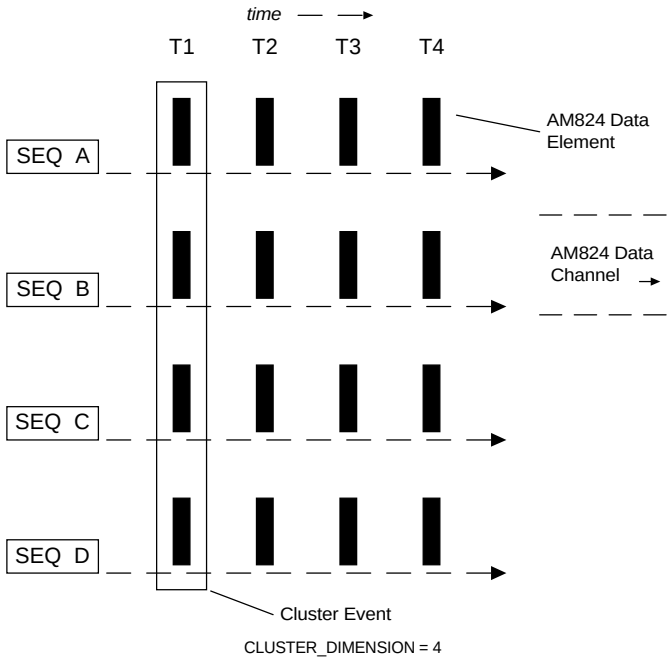

Figure A.3 Sequences and Cluster Events

A.3.1 Clusters and MPX-MIDI Data Channels

Clusters are often used to combine two or more AM824 Data Channels of audio or MIDI data. This ensures that the audio and/or MIDI data is synchronized. However, this also requires that audio and MIDI Data Channels use the same sample rate. Audio data generally requires much higher bandwidth than MIDI data. To conserve IEEE1394 bandwidth and system resources, it is desirable to combine several MIDI streams into a single AM824 Data Channel.

The MPX-MIDI Data Channel mechanism is provided for this purpose. As noted above, each MIDI data stream may directly contain up to 16 MIDI channels. Each complete MIDI data stream is carried by a particular MPX-MIDI Data Channel, which acts as a kind of “virtual MIDI cable”. Several MPX-MIDI Data Channels are multiplexed into a single MIDI Conformant Data Channel (a type of AM824 Data Channel).

The Data Block Count (*DBC*) field within the CIP header is used to specify the MPX-MIDI Data Channel Number associated with each MIDI event within the CIP. The lower 3 bits of the *DBC* field indicate 1 of 8 possible MPX-MIDI Data Channels. All MIDI events within a given Cluster use the same MPX-MIDI Data Channel Number. When a CIP contains multiple cluster events, the *DBC* value indicates the sequence number for the first Data Block (cluster event) within the CIP. The effective *DBC* value is incremented by 1 for each successive cluster event (Data Block) within the CIP.

When all eight MPX-MIDI Data Channels are used, a single MIDI Conformant Data Channel can carry a total of 128 MIDI channels (16 MIDI channels for each MIDI data stream). If more MIDI channels or data streams are needed, two or more MIDI Conformant Data Channels may be used.

Often, a particular MPX-MIDI Data Channel may not actually be used. For example, a source application may only provide one or two MIDI data streams (leaving six or seven MPX-MIDI Data Channels unused). In this case, the transmitter should send an 80H “No Data” code on the unused MPX-MIDI Data Channels. (If the CIP contains only MIDI Conformant data and there is no actual MIDI data to send, the CIP should be an empty packet rather than a packet containing only “No Data” codes.)

A set of MPX-MIDI Data Channels carried by a particular MIDI Conformant Data Channel may optionally be interleaved with other AM824 Data Channels to form a sequence of cluster events placed into CIP packets, as described in [4] and in this Annex.

Figure A.4 shows how 8 MPX-MIDI Data Channels are multiplexed across a series of Clusters (packaging into distinct CIPs is not shown).

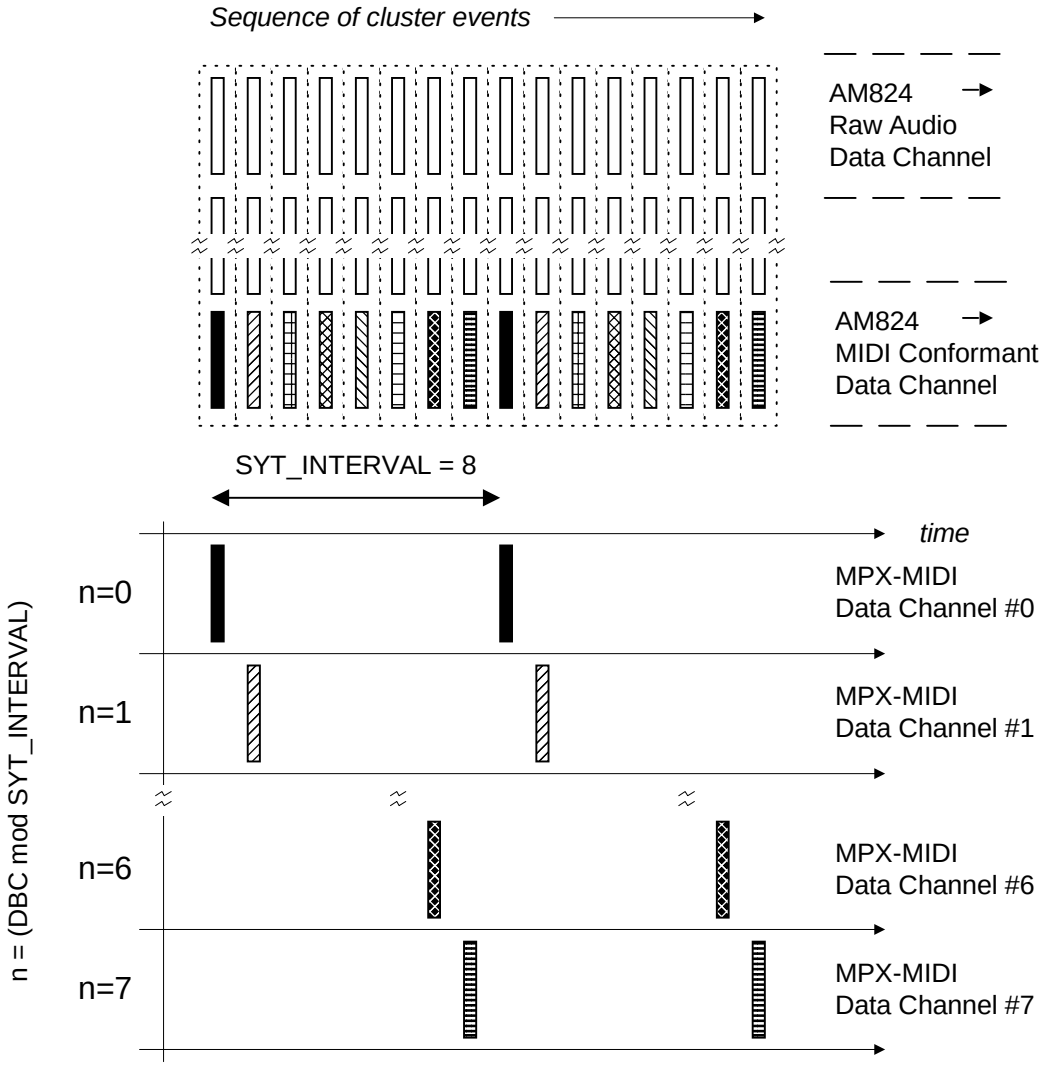

Figure A.4 Clusters, AM824 Data Channels and MPX-MIDI Data Channels

A.3.2 Timing for Audio and MIDI Data

Timing information for audio and MIDI data is handled differently. For audio data, each CIP must contain multiple Clusters so that a number of audio samples can be transmitted for each stream during each 125 microsecond frame interval. For example, a 32 kHz sample rate typically uses four Clusters per CIP, while a 48 kHz sample rate uses six Clusters per CIP. Each successive Cluster of audio events occurs at a different time, in order to provide an even flow of samples at the desired audio sample rate. Calculation of event presentation time is somewhat complicated for several reasons: (a) an explicit time stamp (*SYT*) is only provided every 8 or 16 Clusters, according to *SYT_INTERVAL*; (b) the number of Clusters per CIP is not the same as *SYT_INTERVAL*, so that *SYT* might apply to any of the Clusters in a given CIP; (c) for *SFC* = 44.1 kHz, the number of Clusters per packet is variable. More details on timing and time stamps are given in the preceding section.

Timing for MIDI data is simpler. All MIDI events in a given CIP share the same *SYT* time stamp, even when multiple Clusters are used. Note that the order of events in a given MIDI data stream must be strictly preserved in order to produce the intended result, even (especially!) when several events share the same time stamp. If a given CIP contains an invalid *SYT*, the receiver should use the most recently received valid *SYT*. The nominal timing accuracy for MIDI data depends on *SFC*:

SFC	Interval between valid *SYT*	Nominal MIDI timing accuracy
32 kHz	250 microseconds	+/- 125 microseconds
44.1 kHz	~ 181 microseconds	+/- 125 microseconds **
48 kHz	~ 166 microseconds	+/- 83 microseconds
96 kHz	~ 166 microseconds	+/- 83 microseconds

** valid *SYT* not present in every CIP

Figure A.5 Nominal MIDI Timing Accuracy

Figure A.6 shows a single CIP containing two Raw Audio Data Channels and two MIDI Conformant Data Channels, at a nominal sample rate of 48 kHz. Since the basic IEEE1394 frame rate is 8 kHz, the CIP contains six Clusters in order to support throughput of 48K events per second per Data Channel. The *time stamp index* (calculated per [A.2]) indicates that the *SYT* time stamp (T0) corresponds to the third Cluster within the CIP. While each Cluster of audio events occurs at a different time, all MIDI events within the CIP occur at the same time, regardless of which Cluster contains them. (As noted above, event ordering within a given MIDI data stream must be strictly preserved.)

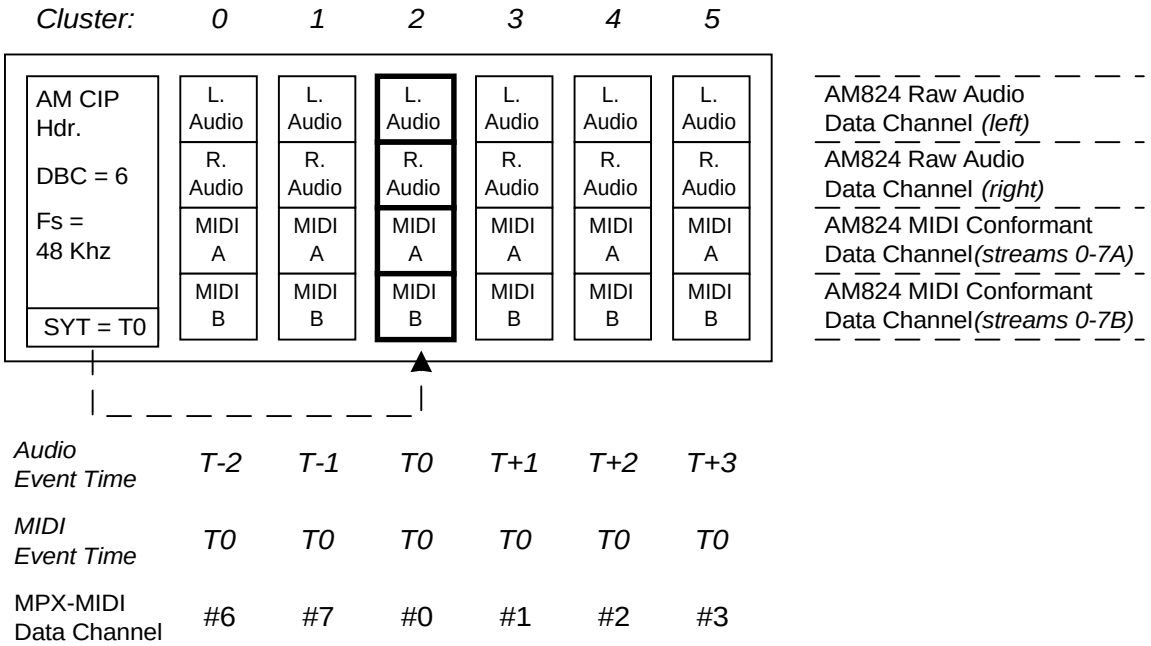
Cluster: 0 1 2 3 4 5
AM CIP Hdr.
DBC = 6
Fs = 48 Khz
SYT = T0
L. Audio
R. Audio
MIDI A
MIDI B
AM824 Raw Audio Data Channel (left)
AM824 Raw Audio Data Channel (right)
AM824 MIDI Conformant Data Channel(streams 0-7A)
AM824 MIDI Conformant Data Channel(streams 0-7B)
Audio Event Time T-2 T-1 T0 T+1 T+2 T+3
MIDI Event Time T0 T0 T0 T0 T0 T0
MPX-MIDI Data Channel #6 #7 #0 #1 #2 #3

Figure A.6 CIP Containing Multiple Clusters of AM824 Data

Within the CIP shown in Figure A.6, the two MIDI Conformant Data Channels are labeled A and B. Each MIDI Conformant AM824 Data Channel carries eight distinct MPX-MIDI Data Channels. Thus, the two MIDI Conformant Data Channels (A and B) carry a total of 16 MIDI data streams. These streams are identified in two sets: A.0–A.7 and B.0–B.7 (e.g. A.0, A.1, A.2, A.3 ... A.7, B.0, B.1, B.2, B.3 ..... B.7). The letter denotes the MIDI Conformant Data Channel containing a given MPX-MIDI Data Channel, while the number denotes the specific MPX-MIDI Data Channel within the MIDI Conformant Data Channel.

A.3.3 Bandwidth

The bandwidth available for one MPX-MIDI Data Channel is given by

AM824 LABEL = 81H:
1 * 8 [bits] * Fs / 8 (MULTIPLEX_NUMBER) [kbit/sec]

AM824 LABEL = 82H:
1 * 16 [bits] * Fs / 8 (MULTIPLEX_NUMBER) [kbit/sec]

AM824 LABEL = 83H:
1 * 24 [bits] * Fs / 8 (MULTIPLEX_NUMBER) [kbit/sec]

Where MULTIPLEX_NUMBER = 8
(MULTIPLEX_NUMBER := the number of MIDI data streams multiplexed within a single MIDI Conformant Data Channel)

For example, when Fs is 32kHz and only quadlets with LABEL = 81H is used, the bandwidth of each MPX-MIDI Data Channel (and the corresponding MIDI data stream) is 1.28 times greater than the bandwidth of a MIDI 1.0 data stream. (The raw transmission rate of a MIDI 1.0 data stream is 31.25 kbits/sec [for 10-bit elements including 2 framing bits]. The valid bit rate of a MIDI 1.0 data stream is 25 kbit/sec [for 8 bit message data].)

Note that “Enhancement to Audio and Music Data Transmission Protocol Version 1.0” states that the SFC table may vary according to the AM824 LABEL. This means that *Nominal_Sampling_Frequency* denoted in the SFC field may specify a different sampling frequency from that defined in AM protocol Version 1 .0. The enhanced AM protocol includes additional AM824 elements for defining a new SFC table.

(End of Document)